Semantic Matching of Open Texts to Pre-scripted Answers in Dialogue-Based Learning

Semantic Matching of Open Texts to Pre-scripted Answers in Dialogue-Based Learning

Ștefan Rușeți[1], Raja Lala[2], Gabriel Guțu-Robu[1], Mihai Dascălu[1(✉)], Johan Jeuring[2], and Marcell van Geest[2]

[1] Computer Science Department, University Politehnica of Bucharest, Bucharest, Romania
{stefan.ruseti, gabriel.gutu, mihai.dascalu}@cs.pub.ro
[2] Computer Science Department, Utrecht University, Utrecht, The Netherlands
{r.lala, j.t.jeuring}@uu.nl, marcell@marcell.nl

Abstract. Gamification is frequently employed in learning environments to enhance learner interactions and engagement. However, most games use pre-scripted dialogues and interactions with players, which limit their immersion and cognition. Our aim is to develop a semantic matching tool that enables users to introduce open text answers which are automatically associated with the most similar pre-scripted answer. A structured scenario written in Dutch was developed by experts for this communication experiment as a sequence of possible interactions within the environment. Semantic similarity scores computed with the SpaCy library were combined with string kernels, WordNet-based distances, and used as features in a neural network. Our experiments show that string kernels are the most predictive feature for determining the most probable pre-scripted answer, whereas neural networks obtain similar performance by combining multiple semantic similarity measures.

Keywords: Answer matching · Semantic similarity · Natural Language Processing · Neural network

1 Introduction

Serious games incorporated in various learning environments are usually aimed at stimulating users' creativity, as well as their engagement. However, most games frequently use pre-scripted interactions that require the specific selection of one option from a list of predefined potential candidates or actions; in return, this approach limits players' immersion and cognition. Our aim is to address this limitation by enabling learners to type free input answers, that are afterwards mapped onto existing alternatives defined within the game.

This study explores different Natural Language Processing (NLP) techniques for matching free-text student responses to pre-scripted answers in a Dutch serious game. The game is based on a communication scenario in which a player converses with a virtual character throughout a simulation. The entire scenario is scripted by an expert as a sequence of potential interactions and questions that form a decision tree with

S. Isotani et al. (Eds.): AIED 2019, LNAI 11626, pp. 242–246, 2019.
https://doi.org/10.1007/978-3-030-23207-8_45

branches corresponding to pre-scripted answers [1]. Instead of selecting a reply from a list of predefined answers given the sequence of questions, users are now encouraged to write their responses, thus providing them with freedom in writing their own responses; in return, these are mapped to the pre-scripted scenario answers.

Nevertheless, we must emphasize from the beginning a limitation of the matching process, namely that both players' and pre-scripted answers are short [2], which in return limits the performance of some NLP methods. Our aim is to explore different semantic relatedness methods and potential manners in which they can be effectively combined in order to best match responses and augment the existing rule-based system incorporated in most serious games.

The problem tackled in this paper is similar to an answer selection task in question answering, if we consider the candidate replies from our scenario as the possible answers. Several datasets exist for English that cover different versions of this problem, like SQuAD [3], MCTest [4], or InsuranceQA [5]. However, these datasets are significantly larger and the complex deep learning models that obtain the best results on these tasks cannot be applied in our case. Thus, we focused mostly on unsupervised methods.

2 Method

Our dataset was gathered in guided sessions with students who played our serious game and provided free-text inputs throughout their gameplay. In addition, players were given the list of pre-scripted answers after providing their text inputs, as well as a "no match" option when their answer was unrelated to any pre-scripted alternative and were asked to select the option which was closest to their answer. The user inputs contained 52.34 characters/9.84 words on average, while pre-scripted answers were similar in size, but still short having limited contextual information: 59.33 characters and 10.44 words on average. Two experts annotated each student's answer by matching themselves all responses to the closest corresponding pre-scripted answers from a semantic point of view. There were 1,143 evaluations overall, out of which 974 cases were kept based on a majority agreement criterion (i.e., two or more people agree out of the initial players and the two experts). These items were used in the experiments that follow. We ran a two-way random effects model of ICC and Cronbach's alpha which denoted acceptable agreement (Cronbach's alpha of .777) and a high average ICC measure of .742.

The following splitting procedure was used. We considered the two most-answered questions and the two least-answered questions to be outliers and put them in the training set. We were left with 20 questions which were ordered by the number of matching items. We assigned consecutive groups of five questions randomly to training (3), testing (1), and validation (1), thus resulting in a dataset with 12 training, 4 testing and 4 validation questions, each set having a significant number of matching items.

We considered several semantic models in order to maximize the matching process. First, SpaCy (https://spacy.io) is an advanced NLP framework written in Python, which contains a very fast syntactic parser designed for production usage. It incorporates pre-trained Dutch semantic models for part-of-speech tagging and dependency parsing. SpaCy computes similarity scores based on the cosine similarity of average word

vectors of two texts. Second, WordNet is a lexicalized ontology whose Dutch version, the Open Dutch WordNet, contains more than 115,000 *synsets* (i.e., sets of similar words) and corresponding relationships [6]. Semantic distances between words available for Dutch include path length [7], the Leacock-Chodorow and the Wu-Palmer methods [8].

Third, string kernels compute a similarity between two texts by counting common character *n*-grams, without the need for any language-specific tools. This method performs well when comparing texts without the need for a large training set [9]. Different scores can be computed by varying the size of the *n*-grams or by changing the way the sum is computed. The most common types of string kernels are *presence*, *intersection* and *spectrum* [10], each representing different ways of computing character *n*-grams overlap. When evaluated as a single method, we computed the average of the three types of string kernels for *n*-grams ranging in size between 3 to 7 characters.

Given the scores computed with each individual method described above, one possible way of improving the performance of the system is to compute an aggregate score. We implemented a neural network (NN) with one hidden layer that computes the best combination of scores. Several experiments were performed on the validation set to select the most relevant features and hyper-parameters of the network. The network receives as input in the training phase two pairs containing a candidate answer and a given answer, one being a positive match, the other negative (either it matches another candidate or doesn't match anything). The network computes a score for each pair and learns to separate them as much as possible. While considering string kernels as features for the neural network, we computed each of the three types with different values for *n*-gram sizes, namely: 2–3, 4–5, 6–7, and 8–9.

3 Results

Given the matches annotated by experts, we evaluate the performance of each method based on the following three types of accuracy: (a) accuracy when a pre-scripted answer is matched (1-match) – 147 out of 224 input texts; (b) accuracy for not matching any pre-scripted answer (no match) – 77 out of 224 input texts; and (c) global accuracy. Results on the test data are presented in Table 1. The neural network combination was trained on both the training and validation partitions after selecting the best configuration on the validation dataset. The threshold used to determine if there is a match was selected based on the validation data.

The neural network combination obtained the highest overall score, but with only a small improvement compared to the String Kernels method (only one more correct example), which seems to be the best method for this task, by far. One possible explanation for the success of the String Kernels is its ability to detect common keywords (in different forms) in the two texts, while not being influenced by the other words in the sentence. All the other methods take into account all the words in the text by using an average over individual word pairs. String Kernels also have the advantage of working at character-level, thus being more suitable to cases when users provide short answers.

Table 1. Accuracies for the semantic methods applied on the test data.

Method	1-match	No match	Global accuracy
SpaCy	(38/147) 26%	(77/77) **100%**	(115/224) 51%
WN path length	(25/147) 17%	(74/77) 96%	(99/224) 44%
WN Leacock Chodorow	(19/147) 13%	(74/77) 96%	(93/224) 42%
WN Wu-Palmer	(19/147) 13%	(76/77) 99%	(95/224) 42%
String kernels	(72/147) **49%**	(64/77) 83%	(136/224) 61%
Average of SpaCy and string kernels	(33/147) 22%	(77/77) **100%**	(110/224) 49%
Neural network	(72/147) **49%**	(65/77) 84%	(137/224) **61%**

In general, direct subword matching is advantageous in cases where the testing domain has a different lexical distribution to the background data used to develop a matching model (e.g., word embedding data). It is likely that partitioning the data set based on scenario questions may have had an effect on the nature of responses between data partitions, especially in terms of word overlap. Moreover, the validation set is more skewed towards *No match* cases, which in turn may bias methods tuned on this set. With these factors in mind, it appears that methods that don't rely on prior knowledge perform better on the test set. For a general-purpose matching method, string kernels proved to be the best option between the selected methods. However, there is clearly still quite a lot of room for improvement.

4 Conclusions

This paper describes the research of text-matching methods for mapping open text input to predefined scripted dialogue response options. We implemented a number of domain-independent text-matching methods including WordNet semantic distances and string kernels, as well as corpora dependent methods (i.e., spacy models). We evaluated these alongside a neural network which integrates our text matching scorers. Overall, the NN combination method achieved the best performance on our test data. However, its performance was quite close to string kernels. Given the additional overhead required to train and run the NN, it appears that string kernels are the best option for integrating a generic, domain independent text-matcher into a serious game.

Iterative improvement of the dialogue design using text analysis methods appears to be a promising way to help ensure open text inputs are dealt with appropriately. For example, shaping the dialogue to encourage user responses to be more specific to the topic discussed will likely make matching easier and semantic models more useful.

We also expect that incorporating more dialogue context into text matching methods may be beneficial when enough consistent user data is available. In this case, we could make more use of the sequence-to-sequence methods that drive many conversational AI chatbots, without sacrificing control of the dialogue structure.

Acknowledgments. This activity has received funding from the European Institute of Innovation and Technology (EIT). This body of the European Union receives support from the European Union's Horizon 2020 research and innovation programme. This research was also partially supported by the 644187 EC H2020 RAGE project.

References

1. Jeuring, J., et al.: Communicate! — a serious game for communication skills —. In: Conole, G., Klobučar, T., Rensing, C., Konert, J., Lavoué, É. (eds.) EC-TEL 2015. LNCS, vol. 9307, pp. 513–517. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24258-3_49
2. Holtgraves, T., Han, T.L.: A procedure for studying online conversational processing using a chat bot. Behav. Res. Methods **39**(1), 156–163 (2007)
3. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250 (2016)
4. Richardson, M., Burges, C.J.C., Renshaw, E.: MCTest: a challenge dataset for the open-domain machine comprehension of text. In: Conference on Empirical Methods in Natural Language Processing (EMNLP 2013), Seattle, Washington, USA, pp. 193–203. ACL (2013)
5. Feng, M., Xiang, B., Glass, M.R., Wang, L., Zhou, B.: Applying deep learning to answer selection: a study and an open task. In: 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), pp. 813–820. IEEE (2015)
6. Postma, M., van Miltenburg, E., Segers, R., Schoen, A., Vossen, P.: Open Dutch WordNet. In: Global WordNet Conference, p. 300, January 2016
7. Mihalcea, R., Corley, C., Strapparava, C.: Corpus-based and knowledge-based measures of text semantic similarity. In: 21st International Conference on AAAI, Boston, Massachusetts, vol. 1, pp. 775–780. AAAI Press (2006)
8. Budanitsky, A., Hirst, G.: Evaluating WordNet-based measures of lexical semantic relatedness. Comput. Linguist. **32**(1), 13–47 (2006)
9. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.: Text classification using string kernels. J. Mach. Learn. Res. **2**, 419–444 (2002)
10. Ionescu, R.T., Popescu, M., Cahill, A.: Can characters reveal your native language? A language-independent approach to native language identification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1363–1373 (2014)